Satisfiability techniques for computing minimal tie sets in reliability assessment

Margaux Duroeulx, Nicolae Brinzei, Marie Duflot, Stephan Merz

SATISFIABILITY TECHNIQUES FOR COMPUTING MINIMAL TIE SETS IN RELIABILITY ASSESSMENT*

MARGAUX DUROEULX[1,2], NICOLAE BRINZEI[1],
MARIE DUFLOT[2], STEPHAN MERZ[2]

[1] *University of Lorraine, CNRS, CRAN, F-54000 Nancy, France*
[2] *University of Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France*

Estimates of system reliability crucially rely on qualitative techniques for determining the impact of component failures. Formally, the structure function of a system determines minimal tie or cut sets that are instrumental for quantitative techniques of reliability assessment. This paper describes three techniques, based on Boolean satisfiability solving, for computing minimal tie sets.

Keywords: Hasse diagram, reliability, tie set, cut set, satisfiability.

1. Introduction

A complex system consists of many components that interact with each other, such as a hydroelectric dam or a nuclear plant. The reliability of such systems is typically assessed using probabilistic methods, taking into account the probabilities of failures of individual components. The impact of component failures on the status of the entire system (*operating* or *failed*) is typically represented graphically, using fault trees, reliability block diagrams or binary decision diagrams. Mathematically, this dependency is described by the *structure function* [4], which can be expressed as a Boolean formula using logical connectives $\wedge$ (and), $\vee$ (or), $\neg$ (not), as well as derived connectives such as k-out-of-n ($koon$). In particular, this function determines *tie sets* and *cut sets*, which can be organized in a Hasse diagram; minimal tie (or cut) sets are required for assessing reliability [2]. Binary Decision Diagrams (BDDs) are a well-known canonical representation of Boolean functions, and several techniques in reliability analysis rely on them [6]. As an alternative to BDDs, efficient techniques and tools for propositional satisfiability (SAT) solving, based on clause representations

*This work was partly supported by the French PIA project "Lorraine Université d'Excellence", reference ANR-15-IDEX-04-LUE.

of Boolean formulas, have been developed over the past two decades [1]. In this paper we present three methods for computing minimal tie sets that rely on these techniques. Our methods differ in the format that they expect the structure function to be expressed in; in particular, our second method relies on a representation in conjunctive normal form (CNF) that underlies state-of-the-art SAT solvers.

Organization of the paper. Section 2 introduces the underlying concepts. Section 3 presents an approach for reliability assessment. Our three methods for computing minimal tie sets are described in sections 4–6, section 7 illustrates the third approach, and section 8 concludes the paper.

2. Notations

The status of components and systems is represented using Boolean variables, where 1 (0) means that the component or system is operating (failed). The configuration of a system with n components can thus be represented as a tuple $\langle x_1, \ldots, x_n \rangle$ of bits. The set $\mathcal{C}$ of configurations is endowed with a partial order that is defined componentwise: for $x = \langle x_1, \ldots, x_n \rangle$ and $y = \langle y_1, \ldots, y_n \rangle$, we write $x \preceq y$ if $x_i \leq y_i$ holds for all $1 \leq i \leq n$. The configurations $\vec{0}$ and $\vec{1}$ (where no, respectively all, components work) are the smallest and largest elements of the ordered set of tuples.

Structure function. The *structure function* $f : \mathcal{C} \to \{0, 1\}$ indicates the state of the system, given a configuration of its components. It can be expressed as a Boolean formula, and we consider two normal forms of such formulas: a formula is in conjunctive normal form (CNF) if it has the shape $\bigwedge_{i=1}^{p} \bigvee_{j=1}^{q_i} l_{ij}$, and it is in disjunctive normal form (DNF) if it is written as $\bigvee_{i=1}^{p} \bigwedge_{j=1}^{q_i} l_{ij}$, where the l_{ij} are literals (variables or their negations).

A system is *coherent* if the structure function f of the system is monotonous, i.e. $f(x) \leq f(y)$ whenever $x \preceq y$. Note that the structure function of a coherent system can be represented as a negation-free formula. A system is *non-trivial* if the structure function is not constant. If f is the structure function of a coherent, non-trivial system, then $f(\vec{0}) = 0$ and $f(\vec{1}) = 1$. In the following, we restrict our attention to such systems.

Hasse diagram. The order relation on the set $\mathcal{C}$ of configurations can be represented as a Hasse diagram whose nodes are configurations. Node x is a *father* of node y, and y is a *son* of x, if $y \preceq x$ and if for all z such that $y \preceq z$ and $z \preceq x$, either $z = y$ or $z = x$. The *ancestor* relation in the Hasse diagram is the reflexive-transitive closure of the father relation; it corresponds to the order $\preceq$.

Tie sets and cut sets. A *tie set* (*cut set*) is a set of system components whose simultaneous functioning (failure) leads to a proper functioning (failure) of the system. A *minimal tie set* (*minimal cut set*) is a tie set (cut set) which does not contain any other tie set (cut set). We identify tie sets (cut sets) and the corresponding configurations: x is a tie set if $f(x) = 1$, and a cut set if $f(x) = 0$. Hence, a tie set x is minimal if $f(y) = 0$ for all $y \prec x$, and a cut set x is minimal if $f(y) = 1$ for all $y \succ x$. For a coherent system, a tie set is minimal iff all its sons in the Hasse diagram are cut sets, and a cut set is minimal if all its fathers are tie sets.

3. Reliability assessment

The reliability function R computes the reliability of the system from the reliabilities R_i of the components c_i. Given the minimal tie sets, an approach from [2] computes the reliability function of the system. A weight equal to 1 is associated to each minimal tie set and is propagated from it to its ancestors (the weight is in the upper right corner of a node, Figure 1). The obtained weight of a node shows how many times each ancestor has been counted taking only the minimal tie sets into consideration. (added to a monomials set with positive contribution). Then all the nodes whose weight is equal to 1 are removed from the Hasse diagram. In the remaining subgraph, the weight of a lower node is reduced to 1 so it is counted only once. This weight reduction is propagated to all the ancestors of the node, indicating how often the monomial corresponding to this node must be subtracted in the reliability function (added to a monomials set with negative contribution). The process is repeated iteratively until all nodes are counted only once and the relevant monomials are included (positively or negatively). For Figure 1, the reliability function is $R = R_1 + R_2 + R_3 - R_1 \cdot R_2 - R_1 \cdot R_3 - R_2 \cdot R_3 + R_1 \cdot R_2 \cdot R_3$.

Since minimal tie sets are needed, this paper suggests methods to compute them effectively.

4. Computing minimal tie sets from a DNF representation

When the structure function f is given in DNF, $f = \bigvee_{i=1}^{p} \bigwedge_{j=1}^{q_i} l_{ij}$,[a] each term $\bigwedge_{j=1}^{q_i} l_{ij}$ corresponds to a tie set $\langle x_1, \ldots, x_n \rangle$ where $x_k = 1$ if and only if the variable x_k is among the literals l_{ij}. Moreover, for any tie set

[a]Since we consider only coherent systems, we may w.l.o.g. assume that only positive literals appear.

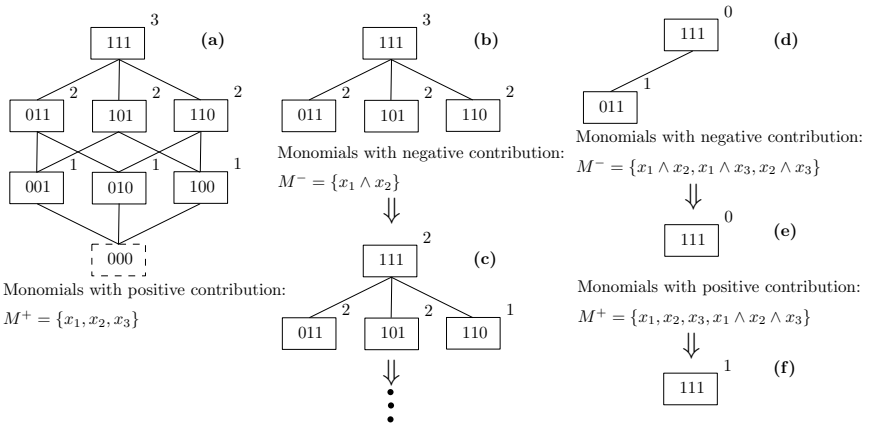

Figure 1. Reliability assessment of a 1oo3 system by means of Hasse diagram. Minimal tie sets are 001, 010, and 100.

y of the system there exists a term in the DNF such that $x \preceq y$ holds for the configuration x corresponding to that term. It then only remains to compute the minimal tie sets. Our algorithm takes as input a set S of configurations and computes the subset of S that contains the minimal configurations w.r.t. the order $\preceq$. Applied to the set T of tie sets obtained from the terms of the DNF, the algorithm therefore computes all minimal tie sets of the system (from the structure function of the system in DNF).

5. Computing minimal tie sets from a CNF representation

Although the algorithm of section 4 is very simple, it expects the structure function to be represented in DNF format. Converting an arbitrary propositional formula to DNF format leads to an exponential blow-up in general and is therefore not practical. SAT solvers usually expect their input to be presented in CNF format, for which there exist algorithms that produce a CNF representation that is linear in the length of the original formula, at the expense of introducing additional propositional variables.

When the structure function is given in conjunctive normal form (CNF), we will employ a SAT solver such as MiniSat [5] for producing a set T of tie sets, until every tie set x of the system is covered by the tie sets in T, in the sense that $y \preceq x$ holds for some $y \in T$. The algorithm of section 4 is then applied to T for obtaining the set of minimal tie sets.

Given a CNF formula f, the SAT solver decides whether f is satisfiable

and, if so, produces a model of f, represented as a set of literals whose conjunction implies f. In our application, a model corresponds to a tie set. We can obtain more ties by adding the disjunction of the negated literals to the original input formula and calling the SAT solver again. In fact, modern SAT solvers are *incremental* in the sense that new clauses can be added on the fly, and intermediate results computed during previous calls are maintained. The procedure is repeated until the SAT solver determines the input formula to be unsatisfiable.

Example 5.1. Consider a system of 6 components whose structure function is given by $f = x_1 \wedge (x_2 \vee x_3) \wedge (x_3 \vee x_4) \wedge (x_5 \vee x_6)$.

If the first model generated is the tuple $\langle 1, 1, 0, 1, 0, 1 \rangle$, corresponding to the tie set $\{x_1, x_3, x_4, x_6\}$, the clause $\neg x_1 \vee \neg x_3 \vee \neg x_4 \vee \neg x_6$ is added to the formula, and the subsequent call to the SAT solver must produce a different model. Continuing this way, we obtain all the minimal tie sets of the system: $\{\{1, 3, 6\}, \{1, 3, 5\}, \{1, 2, 4, 5\}, \{1, 2, 4, 6\}\}$. □

6. From minimal cut sets to minimal tie sets

In reliability theory, minimal cut sets are frequently obtained based on a fault tree and its structure function. Cut sets can directly be read off a representation of the structure function in CNF format, dually to how a DNF representation yields tie sets, and minimal cut sets can be obtained in a manner analogous to the computation of minimal tie sets in section 4.

We now present an algorithm for computing minimal tie sets from minimal cut sets. The inputs to the algorithm are the structure function f of the system and the set *MinCut* of minimal cut sets. It returns the set of minimal tie sets. Algorithm 1 below is based on following the arcs in the Hasse diagram. We observe that the fathers of minimal cut sets are tie sets, and the algorithm then follows "son" links in the diagram until finding minimal tie sets. The intuition is that minimal tie sets are often at a small distance from minimal cut sets and that therefore few arc traversals are necessary for computing them. The algorithm relies on the auxiliary functions *fathers* and *sons* that, given a node, return its father and son nodes in the Hasse diagram, respectively.

Theorem 6.1. *Assume that f is the structure function of a coherent and non-trivial system, whose set of minimal cut sets is given by MinCut. Then Algorithm 1 computes the corresponding set of minimal tie sets.*

Algorithm 1 Compute the minimal tie sets from the minimal cut sets

Require: f: structure function
Require: $MinCut$: set of minimal cut sets
Ensure: $MinTie$: set of minimal tie sets
  $Tie \leftarrow \bigcup \{ fathers(c) : c \in MinCut \}$
  $MinTie \leftarrow \emptyset$
  while $Tie \neq \emptyset$ **do**
    take $t \in Tie$
    $Tie \leftarrow Tie \setminus \{t\}$
    $ts \leftarrow \{s \in sons(t) : f(s) = 1\}$
    if $ts = \emptyset$ **then**
      $MinTie \leftarrow MinTie \cup \{t\}$
    else
      $Tie \leftarrow Tie \cup ts$
    end if
  end while
  return $MinTie$

Proof. The algorithm maintains the following loop invariant:[b]

(1) The sets Tie and $MinTie$ contain tie sets, resp. minimal tie sets.
(2) For any minimal tie set x of the system described by f, there exists some $y \in Tie \cup MinTie$ such that $x \preceq y$. □

Optimization. Algorithm 1 may handle the same tie set repeatedly. This can easily be avoided by adding a variable $TieSeen$ that contains all tie sets that have already been considered. $TieSeen$ is initialized to Tie. In the **else**-branch of the loop body, $(ts \setminus TieSeen)$ is added to Tie and the set ts is added to $TieSeen$. The correctness proof is easily adapted.

7. Reliability assessment

We illustrate the use of Algorithm 1 by means of an example due to Rogova et al. [7]. The system consists of one main controller (MC) and two channels. Each channel is made up of a brake controller (BC), a sensor (S) and a braking system which is the actuator (BS). The set of components is thus $\{MC, BC_1, S_1, BS_1, BC_2, S_2, BS_2\}$.[c]

[b] A full proof appears in Appendix A.
[c] We assume that the diagnostic cannot fail, therefore it is not considered as a component.

The architecture follows the 1oo2D (1oo2 with diagnostic) style [3]: both channels are used as long as they both work, but in the case of a fault signal from any one of the two sensors, the system will switch to the other channel. The structure function (in CNF format) is given as

$$\begin{array}{llllll} \wedge & MC & & & & \\ \wedge & (BC_1 \vee BC_2) & \wedge & (BC_1 \vee S_2) & \wedge & (BC_1 \vee BS_2) \\ \wedge & (S_1 \vee BC_2) & \wedge & (S_1 \vee S_2) & \wedge & (S_1 \vee BS_2) \\ \wedge & (BS_1 \vee BC_2) & \wedge & (BS_1 \vee S_2) & \wedge & (BS_1 \vee BS_2). \end{array}$$

From this presentation, we can directly read off the minimal cut sets

$$\begin{array}{l} \{MC\},\ \{BC_1, BC_2\},\ \{BC_1, S_2\},\ \{BC_1, BS_2\},\ \{S_1, BC_2\}, \\ \{S_1, S_2\},\ \{S_1, BS_2\},\ \{BS_1, BC_2\},\ \{BS_1, S_2\},\ \{BS_1, BS_2\}. \end{array}$$

Algorithm 1 computes two minimal tie sets $\{MC, BC_1, S_1, BS_1\}$ and $\{MC, BC_2, S_2, BS_2\}$ and method of section 3 gives the reliability function

$$\begin{aligned} R &= R_{MC} \cdot R_{BC1} \cdot R_{S1} \cdot R_{BS1} \\ &\quad + R_{MC} \cdot R_{BC2} \cdot R_{S2} \cdot R_{BS2} \\ &\quad - R_{MC} \cdot R_{BC1} \cdot R_{S1} \cdot R_{BS1} \cdot R_{BC2} \cdot R_{S2} \cdot R_{BS2}. \end{aligned}$$

Restricting to dangerous failures, the reliability of the MC and BC components is assumed to follow an exponential distribution with parameter $\lambda = 5.5 \cdot 10^{-8} h^{-1}$, the S components an exponential distribution with parameter $\lambda = 1.09 \cdot 10^{-8} h^{-1}$, whereas the BS components follow a Weibull distribution with shape parameter $\beta = 1.77459$, scale parameter $\eta = 8.2942 \cdot 10^4 h$, and location parameter $\tau = 0$.

Figure 2 illustrates the reliability of the system during 20 years, from a *fault tree* analysis and a computation with a *Hasse diagram*.

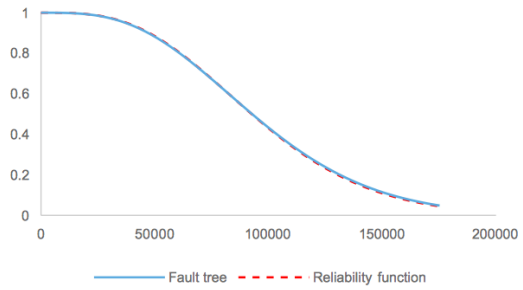

Figure 2. System reliability over time.

The reliability of the system after 20 years (i.e. $1.752.10^5$ hours) is estimated as 0.0483 with a fault tree analysis and 0.0446 with a Hasse diagram analysis. This small difference (1%) can be imputed to computational approximations. Moreover, since we only consider dangerous failures it is not unreasonable to get a reliabilty of 0.99 after 1 year and 0.04 after 20 years.

In this example, Algorithm 1 enabled us to compute the minimal tie sets of the system, then the approach from [2] used these minimal tie sets as inputs for computing the reliability of the system. We thus illustrated that, by coupling our computation of the minimal tie sets with the reliability analysis of [2], we can easily compute the reliability of a system.

8. Conclusion

Probabilistic methods for reliability assessment rely on qualitative analyses of the structure functions, and we suggest that techniques developed in satisfiability theory can be useful for carrying out these analyses. In particular, Algorithm 1 provides an efficient method in practice when minimal cut sets are known. Of course, dual versions of our methods can be used to compute minimal cut sets from CNF or DNF representations, as well as from minimal tie sets. Although we only considered coherent systems in this paper, the method from [2] also applies to non-coherent ones, and we plan to extend our methods to such systems. We also want to address multi-state systems where component and system states are not just Boolean.

References

[1] A. Biere, M. Heule, H. van Maaren, and T. Walsh, editors. *Handbook of satisfiability*, volume 185. IOS Press, 2009.

[2] N. Brinzei and J-F. Aubry. An approach of reliability assessment of systems based on graphs models. *European Safety and Reliability Conference ESREL*, CRC Press/Balkema, Taylor & Francis Group:1485–1493, 2015.

[3] W. M. Goble and H. Cheddie. *Safety Instrumented Systems Verification: practical probabilistic calculations*. ISA, 2004.

[4] A. Kaufmann, D. Grouchko, and R. Cruon. *Mathematical Models for the Study of the Reliability of Systems*, volume 124. Elsevier, mathematics in science and engineering edition, 1977.

[5] Eén Niklas and Sörensson Niklas. An extensible SAT-solver. *6th Intl. Conf. Theory and Applications of Satisfiability Testing*, 2919:502–518, 2003.

[6] A. Rauzy. Mathematical foundations of minimal cutsets. *IEEE Transactions on Reliability*, 50(4):389–396, 2001.

[7] E. Rogova and G. Lodewijks. Braking system redundancy requirements for moving walks. *Reliability Engineering and System Safety*, 133:203–211, 2015.

Appendix A. Proof of Theorem 6.1

Proof. We show that Algorithm 1 maintains the following loop invariant:

(1) The sets *Tie* and *MinTie* contain tie sets, resp. minimal tie sets.
(2) For any minimal tie set x of the system described by f, there exists some $y \in \mathit{Tie} \cup \mathit{MinTie}$ such that $x \preceq y$.

The invariant implies the assertion of the theorem: when $\mathit{Tie} = \emptyset$, the set *MinTie* contains only minimal tie sets (1), and all minimal tie sets (2).

We now show that the invariant holds when the loop condition is evaluated for the first time. Recall that every father of any minimal cut set is a tie set; moreover, $\mathit{MinTie} = \emptyset$ upon the first entry into the loop, establishing condition (1). For condition (2), assume that $x = \langle x_1, \ldots, x_n \rangle$ is a minimal tie set. Since the system is non trivial, at least one x_j must be 1, hence x has a son $x' = \langle x_1, \ldots, x_{j-1}, 0, x_{j+1}, \ldots, x_n \rangle$, and since x is a minimal tie set, x' is a cut set. Therefore, *MinCut* contains a minimal cut set $y' = \langle y'_1, \ldots, y'_n \rangle$ such that $x' \preceq y'$. Moreover, we must have $y'_j = 0$: otherwise, it would follow that $x \preceq y'$ but $f(x) = 1$ and $f(y') = 0$, since x and y' are respectively a tie set and a cut set, contradicting the assumption that the system is coherent. Now consider the father $y = \langle y'_1, \ldots, y'_{j-1}, 1, y'_{j+1}, \ldots, y'_n \rangle$ of y': by construction we have $x \preceq y$ and $y \in \mathit{Tie}$, establishing condition (2).

It remains to show that the invariant is preserved by all executions of the loop body, so assume that it holds upon the entry of the loop and that $\mathit{Tie} \neq \emptyset$, and let t be the element of *Tie* chosen for executing the loop body. We will denote by Tie' and MinTie' the values of these variables at the end of the loop body.

By condition (1), we know that t is a tie set, and ts is a set of tie sets by definition. Also, t is a minimal tie set iff $ts = \emptyset$, and this shows that condition (1) still holds after the execution of the loop body. For condition (2), let again x be a minimal tie set. If $x \preceq y$ holds for some $y \in (\mathit{Tie} \setminus \{t\}) \cup \mathit{MinTie}$ upon the entry to the loop body, we still have $y \in \mathit{Tie}' \cup \mathit{MinTie}'$, and condition (2) is trivially preserved. Now assume that $x \preceq t$. If $ts = \emptyset$, then t is a minimal tie set, hence $t = x$ and we have $t \in \mathit{MinTie}'$, establishing condition (2) at the end of the loop body. Otherwise, t is not minimal and hence different from x. Since $x \preceq t$, there exists a component k that is functioning in $t = \langle t_1, \ldots, t_{k-1}, 1, t_{k+1}, \ldots, t_n \rangle$ but not in x. The tuple $t' = \langle t_1, \ldots, t_{k-1}, 0, t_{k+1}, \ldots, t_n \rangle$ is thus a son of t, and it satisfies $x \preceq t'$. As the system is coherent, t' is a tie and hence $t' \in ts \subseteq \mathit{Tie}'$. □